Approximation Algorithm

Prof. Palash Dey

Department of Computer Science and Engineering

Indian Institute of Technology, Kharagpur

Week – 12

Lecture 57

Lecture 57 : SDP Based Approximation Algorithm for Max Cut

Welcome. So, in the last class we have started semi definite programming, we have done a basic overview of semi definite program. In this class we will see how semi definite programs can be used for designing an approximation algorithm for max cut ok. So, let us start. So, today's problem is max cut problem. So, what is the problem? Let us recall input is an undirected weighted graph $G=(V,E)$ and weights of the edges and output or goal compute a cut is a strict subset of V is not equal to empty set which maximizes weight of the boundary edges of $\delta(S)$ ok.

So, let us see first SDP relaxation of the problem. First let us see as usual the ILP formulation of the problem which is exact but then we will relax it to SDP not LP. So, because we will relax it to SDP and in SDP formulation there are variables indexed by ij and x_{ij} should be equal to x_{ji}. So, the variable matrix should be a symmetric matrix and positive semi definite matrix that is the requirement.

So, let us think of variables as x_{ij}s where ij are vertices. So, the variables are or we can have variables for each vertex i and they will be eventually converted to vectors because in the last class as we have seen these SDPs are equivalent to vector programs. So, we will have a variable y_i for every $i \in V$. Let us assume V to be $\{1,2,\ldots,n\}$, y_i takes value 1 if i belongs to And in standard ILP formulation we typically have variables corresponding to indicator random variables. So, in a standard ILP formulation we will allow y_i to take value 0 if y does not belong to S, but remember that we have to write our objective function in terms of inner products of y_i and y_j which are simple products $y_i \cdot y_j$.

So, that is why it is convenient to make y_i take value -1 if i does not belong to S. So, y takes value 1 if i belongs to S and -1 otherwise ok. Now, with this let us see how I can denote the cut size I which I want to maximize. So, I want if you look at this for an edge $e=\{i,j\} \in E$, I want a function which will take value 1.

which will take value 1 if the edge $\{i,j\}$ belongs to the cut and value 0 if the edge $\{i,j\}$ does not belong to the cut. So, if edge $\{i,j\}$ belongs to the cut if and only if exactly one of i and j belongs to S. In that case you see the value of the product y_i and y_j, $y_i \cdot y_j = -1$. On the other hand if both i and j belongs to S or both i and j does not belong to S, then the value of y i times y j will be 1 and in that case the edge $\{i,j\}$ does not contribute to the cut. So, $w_{ij}(1 - y_i \cdot y_j)$.

So, this function is 1 if $\{i,j\}$ this edge belongs to the cut and 0 otherwise ok and that is it. So, the only condition that we need is y_j or y_i should take value either 1 or -1. So, subject to $y_i \in \{1,-1\}$ for all $i \in [n]$ ok. So, this is the integer linear programming formulation exact formulation. Now, we will relax it.

So, vector programming relaxation because in a vector program these kind of constraints that y_i belongs to some discrete set is not allowed. So, let us see how we can replace it with some condition which are allowed in vector programs. So, vector programming relaxation So, we replace the idea is we replace each y_i by an n dimensional vector v_i n dimensional vector v_i of length i or unit vectors in particular. length 1 ok. So, what is the relaxed vector program maximize $\sum_{e=\{i,j\} \in E} w_{ij}(1 - v_i \cdot v_j)$ subject to v_i are n dimensional vector and their length is 1.

So, that means, their inner product with itself is 1 for all $i \in [n]$ and v_i is an n dimensional real vector ok. So, why this is a relaxation because each v_i you can replace it with a vector where So, each v_i you replace it with an n dimensional vector where the first coordinate is v_i and rest are 0. So, if I replace so, this is the v_i. y_i with this v_i's, then all the constraints getting satisfied the objective function remains same. Hence, this is a relaxed LP, this is a relaxed vector program, this is a relaxation.

vector program opt because it is a relaxation and because we are maximizing this is greater than equal to ILP opt which is same as opt ok. So, we have a vector program now as usual we will solve it and we will do a randomized rounding. So, we solve the vector program in polynomial time and obtain an optimal solution v_i^* $i \in [n]$. Now, they are n dimensional unit vectors. So, these vectors lie in an n dimensional sphere around origin.

So, since $v_i^* \cdot v_i^* = 1$ for all $i \in [n]$ these vectors lie on the unit sphere around origin in the n dimensional Euclidean space R^n. And the idea is we will take a random hyperplane we will take a random hyperplane passing through the origin and this random hyperplane partitions this points which corresponds to vertices and this gives us the cut. So, idea pick random hyperplane passing through the origin such a hyperplane partitions the vertices into $(S, V \setminus S)$ output that partition So, that is the idea that we will take a random hyperplane passing through origin and use it to divide the vertices into 2 groups. Let us see how we can implement it implementing the idea. for that we pick a random vector $r=(r_1,\ldots,r_n)$ by sampling each r_i , $i\in[n]$ from a standard normal distribution $N(0,1)$ mean at 0 and standard deviation is 1. The normal distribution can be simulated how we draw samples this normal distribution using samples from uniform distribution we can sample points from normal distribution. So, let me just highlight it as a fact the normal distribution can be simulated by an algorithm that draws uniform samples from $[0,1]$ only. So, it is a technicality it just says that if we are able to draw uniformly random samples from $[0,1]$, then we can draw a uniform or we can draw a sample from normal standard normal distribution with mean 0 and standard deviation 1.

that is it. So, now, how we partition the vertices? We put a vertex $i\in S$ if $v_i^* \cdot r_i \geq 0$ and $i\in V\setminus S$ otherwise. So, this gives the partition and we simply output that partition. So, let us see why this is a good thing to do and what is the approximation guarantee. So, to analyze this algorithm we need some fact about normal distributions. So, let us write them down without proof.

The proof can be found from any standard book on probability theory. The normalization so, r is the vector chosen from here. So, if I normalize r because you see the a length of r need not be 1. So, let us normalize r the normalization of r which is r by the norm of it which is $r\cdot r$ or let us say norm is uniformly distributed over the n dimensional unit sphere that is one. Another fact is that we need is the projection of r onto any two dimensional plane this gives again like a normal distribution with two parameters.

So, let us see the projection of r onto 2 unit vectors e_1 and e_2 which are independent and orthogonal. orthogonal that means, $e_1\cdot e_2=0$. So, if I project e onto e_1 and e_2 then both of them are independent and normally distributed. The projection are independent and follows the standard normal distribution which is a normal distribution with mean 0 and standard deviation 1 ok. So, this is the fact with these two fact let us prove a important lemma from which the an approximation guarantee is immediate.

So, lemma is the probability that an edge $\{i,j\}$ belongs to the cut is $\frac{1}{\pi} arc\cos(v_i\cdot v_j)$. So, let us prove the lemma So, let us draw the 2 dimensional circle with e_1 and e_2 and v_i and v_j. So, let r' be the projection of r onto the plane spanned by v_i and v_j. Now, let us draw So, suppose this is the circle unit circle and here is v_i and suppose here is v_j, draw two lines one is perpendicular to v_j and another is perpendicular to v_i. So, give some name say A B and this is C D.

Now, let us see when does this an edges $\{i, j\}$ contributes to the cut. So, r' is distributed which is it which follows from this fact that r' is distributed uniformly randomly on the circle. of radius 1 containing v_i and v_j. That means, you look at the plane which contains v_i and v_j plane span by v_i and v_j. these 2 vectors and in that plane you look at the circle around origin of radius 1, then r' is uniformly distributed on the perimeter of the circle.

So, then you see when does So, for what angle means where does r_i sits? So, that v_i and v_j their inner product have different sign. $r' \cdot v_i$ and $r' \cdot v_j$ will have different sign only in let us say in which region. So, if r_i falls in say in this region, then its angle with v_i is a suppose this is theta this angle between v_i and v_j is θ and if r_i' falls both on the right hand side of A B and the and the top of C D, then v_i and v_j have the same sign v_i and v_j the inner product of v_i and v_j with R prime have the same sign. If the r' falls in this region which is top of which is top of C D and right of A D this and same with this regions. So, only if r_i' falls in this region.

is black region then its inner product with v_i and v_j have different sign and they fall in different part same with here only in say black region ok. So, if this is θ then from high school geometry it can be proved that this is θ and this is also θ. So, the probability that $\{i, j\}$ is cut is at is exactly $\frac{2\theta}{2\pi}$ because r' is uniformly distributed on the perimeter which is $\frac{\theta}{\pi}$. So, let us stop here and in the next lecture we will see how θ is connected with v_i v_j and from that how we can get an approximation guarantee of this algorithm ok. So, let us stop here. Thank you.